Preparing to Run on Endeavour

The Endeavour supercomputer comprises two nodes, Endeavour3 and Endeavour4. Each of the two systems includes 896 cores and six terabytes (TB) of global shared memory. See Endeavour Configuration Details for more hardware information.

The nodes use Pleiades front ends (PFEs) and filesystems, and share the Pleiades InfiniBand fabric. They use PBS server pbspl4.

Operating System

As of Dec. 20, 2021, most NAS systems are running the Red Hat Enterprise Linux-based Tri-Lab Operating System Stack (TOSS 3).

However, both of the Endeavour nodes are still running SUSE Linux Enterprise Server 12 (SLES 12). Endeavour will be migrated to TOSS 3 in the near future.

In the meantime, to build an executable to run on Endeavour, you should use a PFE that is still running SLES 12:

- pfe[*24-27*] are currently running TOSS 3. If you use the Pleiades Front End Load Balancer, it will select one of these PFEs.
- pfe[*20-23*] are still running SLES 12. To use a PFE with SLES 12, you must provide its explicit host name instead of using the load balancer. For example:

```
your_local_host% ssh pfe20
pfe20%
```

Note: pfe[*20-23*] will gradually be transitioned to TOSS 3 and become part of the load balancer pool. To find out the operating system of the PFE you are on, do:

```
pfe% grep . /etc/os-release
```

Connecting to Endeavour

To access the systems, submit PBS jobs from the Pleiades PFEs with the server name **`pbspl4`** included in the **`qsub`** command line or PBS script. You cannot log directly into Endeavour3 and 4 from the PFEs unless you have a PBS job running there.

Accessing Your Data

Use your Pleiades home filesystem (/u/*username*) and your Lustre filesystem (/nobackup) for both Pleiades and Endeavour.

For more information on using Lustre filesystems, see Lustre Basics and Lustre Best Practices.

Compiling and Running Your Code

There are no default modules for compilers, Message Passing Interface (MPI) components, or math libraries. To use these modules, you must first load them.

OpenMP Applications

Load an Intel compiler module and use the Intel compiler **-qopenmp** option to build the executable. Since the Cascade Lake processors used in Endeavour3 and 4 support AVX512, you can experiment with different compiler versions and flags for better performance. For example:

```
pfe21% module load comp-intel/2018.3.222
pfe21% ifort -O2 -qopenmp program.f
```

or

```
pfe21% module load comp-intel/2020.4.304
pfe21% ifort -Ofast -ipo -axCORE-AVX512,CORE-AVX2,AVX -xSSE4.2 -qopenmp program.f
```

Set a specific number of OpenMP threads in your PBS job, and use a thread-pinning method to ensure that the threads do not get in the way of each other on the same core or migrate from one core to another. For example:

```
#PBS ...

module load comp-intel/2020.4.304
setenv OMP_NUM_THREADS 28

/u/scicon/tools/bin/mbind.x -cs -t28 -v ./a.out

or

setenv KMP_AFFINITY compact # or: setenv KMP_AFFINITY scatter
./a.out > output

or

setenv KMP_AFFINITY disabled
dplace -x2 ./a.out > output
```

See Process/Thread Pinning Overview for more information about different pinning methods.

Note: The default OMP_STACKSIZE is set to 200 megabytes (MB). If your application experiences segmentation fault (segfault), experiment with increasing the $OMP_STACKSIZE value to see if it helps.

Applications that Require Scientific and Math Libraries

The Intel Math Kernel Library (MKL) is included in Intel compiler modules (versions 11.1 and later). See MKL to learn how to link to MKL libraries. In many cases, the following commands may be sufficient:

```
pfe21% module load comp-intel/2020.4.304
pfe21% ifort -O2 program.f -mkl
```

Note: By itself, the **-mkl** option implies **-mkl=parallel**, which will link to the threaded MKL library. If your application can benefit from using multiple OpenMP threads when the MKL routines are called, you should also set the environment variable **OMP_NUM_THREADS**, as shown in the OpenMP example in the previous section.

If you want to use a non-threaded MKL library, change the **-mkl** option to **-mkl=sequential**.

MPI Applications

HPE's latest Message Passing Toolkit (MPT) library is recommended. Load the library as shown in the following example:

```
pfe21% module load comp-intel/2020.4.304
pfe21% module load mpi-hpe/mpt
pfe21% ifort -O2 program.f -lmpi
```

During runtime, load the Intel compiler and the MPT modules and use **`mpiexec`** to launch the executable:

```
#PBS ...
module load comp-intel/2020.4.304
module load mpi-hpe/mpt

mpiexec -np xx ./a.out > output
```

Running PBS Jobs

Although the Endeavour3 and Endeavour4 nodes use PFEs and Pleiades filesystems, and share the Pleiades InfiniBand fabric for I/O, they use a separate PBS server: pbspl4. Therefore, PBS jobs cannot run across the Endeavour3 and 4 and Pleiades compute nodes.

Each of the Endeavour nodes has 32 sockets, and each socket includes 28 cores and 192 gigabytes (GB) of physical memory. In each socket, PBS has access to 28 cores and 185 GB of memory, which is known as a minimum allocatable unit (MAU). In addition, one of the sockets is set aside for system operations, leaving 31 sockets available for user jobs.

PBS does not allow jobs to share resources in the same socket. However, the amount of resources in a socket that a job will receive is restricted to the amount requested. This restriction is enforced by the Linux kernel **`cgroups`** feature. For example, if you request one core and 100 GB of memory, one socket will be assigned exclusively for your job, but 27 cores and 85 GB of memory of the socket will not be accessible by your job.

When job accounting is enabled, your job will be charged by the number of MAUs assigned exclusively for your job.

The maximum amount of resources a job can request is:

- **Endeavour3:** 868 cores with 5,750 GB are the maximum resources available for PBS jobs.
- **Endeavour4:** 868 cores with 5,750 GB are the maximum resources available for PBS jobs.

Endeavour PBS Queues

The PBS server pbspl4 manages jobs for Endeavour3 and 4 (**`:model=cas_end`**) and the NAS V100 GPU nodes (**`:model=sky_gpu`** and **`:model=cas_gpu`**). Among the queues listed by the command **`qstat -Q @pbspl4`**, you can run jobs on the two systems using the e_normal, e_long, e_vlong, and e_debug queues. You can also find detailed settings of each queue, such as the maximum walltime and the default or minimum ncpus and memory, using the following command line (the e_normal queue is used in the example):

```
pfe% qstat -fQ e_normal@pbspl4
```

PBS Commands

Run PBS commands (such as **qsub**, **qstat**, and **qdel**, etc.) from the PFEs and specify the PBS server, pbspl4. For example:

```
pfe21% qsub -q queue_name@pbspl4 job_script
pfe21% qstat -nu username @pbspl4
pfe21% qstat job_id.pbspl4
pfe21% qdel job_id.pbspl4
```

If you do not normally run jobs on Pleiades compute nodes, and your primary workload is on Endeavour3 and Endeavour4, you may want to consider setting the **PBS_DEFAULT** environment variable to pbspl4. This allows you to simplify the PBS commands, as follows:

```
pfe21% setenv PBS_DEFAULT pbspl4
pfe21% qsub job_script
pfe21% qstat -nu username
pfe21% qstat job_id
pfe21% qdel job_id
```

Job Submission Examples

The following examples demonstrate how PBS allocates resources for your job.

Note: If you do not specify a resource attribute (such as ncpus or mem), a default value for that attribute will be assigned to your job.

- To request using all the resources in two MAUs:

  ```
  pfe21% qsub -I -lselect=2:ncpus=28:mem=185G:model=cas_end -q queue_name@pbspl4
  ```
- To request 56 cores:

  ```
  pfe21% qsub -I -lselect=1:ncpus=56:model=cas_end -q queue_name@pbspl4
  ```

  Your job will be given access to 56 cores across in two sockets and only the default amount of memory (32G).
- To request 1,850 GB of memory:

  ```
  pfe21% qsub -I -lselect=1:mem=1850G:model=cas_end -q queue_name@pbspl4
  ```

  Your job will be given access to 1,850 GB of memory spread in ten sockets and only the default number of cores (one).
- If you do not specify which Endeavour node to run your job, PBS will assign one for you. The following example specifies Endeavour3 and requests 28 cores and 185 GB of memory:

  ```
  pfe21% qsub -I -lselect=host=endeavour3:ncpus=28:mem=185G:model=cas_end -q queue_name@pbspl4
  ```

Job Accounting

The Standard Billing Unit (SBU) rate for using one MAU per hour is 1.31.

As of October 1, 2019, your group's HECC SBU allocation is charged for usage on any HECC resources including Pleiades, Electra, Aitken, and Endeavour. The output from the **acct_ytd** command does not have separate entries for the different systems. To find Endeavour3 and Endeavour4 usage for your group, use the **acct_query** command instead.

For example, to view the statistics of all of your jobs that ran on Endeavour3 on March 10, 2021, type:

```
pfe21% acct_query -c endeavour3 -d 03/10/21 -u your_username -olow
```

See Job Accounting Utilities to learn more about using the **acct_ytd** and **acct_query** tools.

Article ID: 652
Last updated: 20 Dec, 2021
Revision: 34
Running Jobs with PBS -> PBS on Endeavour -> Preparing to Run on Endeavour
https://www.nas.nasa.gov/hecc/support/kb/entry/652/